Pattern Recognition 87 (2019) 226–236

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

A nonlinear dimensionality reduction framework using smooth geodesics

Kelum Gajamannage[a,*], Randy Paffenroth[a], Erik M. Bollt[b]

[a] *Department of Mathematical Sciences, Worcester Polytechnic Institute, Worcester, MA 01609, USA*
[b] *Clarkson Center for Complex Systems Science, Clarkson University, Potsdam, NY 13699, USA*

ARTICLE INFO

Article history:
Received 7 December 2017
Revised 20 June 2018
Accepted 16 October 2018
Available online 19 October 2018

2010 MSC:
57M50
57N16
58D15

Keywords:
Manifold
Nonlinear dimensionality reduction
Smoothing spline
Geodesics
Noisy measurements

ABSTRACT

Existing dimensionality reduction methods are adept at revealing hidden underlying manifolds arising from high-dimensional data and thereby producing a low-dimensional representation. However, the smoothness of the manifolds produced by classic techniques over sparse and noisy data is not guaranteed. In fact, the embedding generated using such data may distort the geometry of the manifold and thereby produce an unfaithful embedding. Herein, we propose a framework for nonlinear dimensionality reduction that generates a manifold in terms of smooth geodesics that is designed to treat problems in which manifold measurements are either sparse or corrupted by noise. Our method generates a network structure for given high-dimensional data using a nearest neighbors search and then produces piecewise linear shortest paths that are defined as geodesics. Then, we fit points in each geodesic by a smoothing spline to emphasize the smoothness. The robustness of this approach for sparse and noisy datasets is demonstrated by the implementation of the method on synthetic and real-world datasets.

1. Introduction

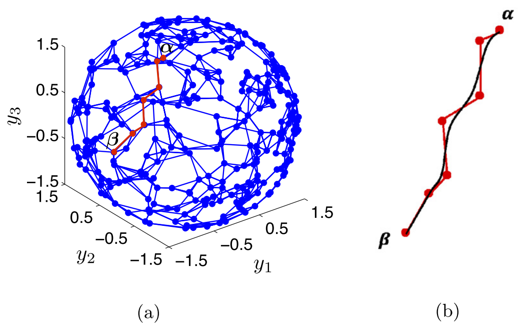
Advanced data collection techniques in today's world require researchers to work with large volumes of nonlinear data, such as global climate patterns [1,2], satellite signals [3,4], social and mobile networks [5,6], the human genome [7,8], and patterns in collective motion [9,10]. Studying, analyzing, and predicting such large datasets is challenging, and many such tasks might be implausible without the presence of Nonlinear Dimensionality Reduction (NDR) techniques. NDR interprets high-dimensional data using a reduced dimension that corresponds to the intrinsic nonlinear dimensionality of the data [11]. Manifolds are often thought of as being smooth, however many existing NDR methods do not directly leverage this important feature. Sometimes, ignoring the underlying smoothness of the manifold can lead to inaccurate embeddings, especially when the data is *sparse* or has been contaminated by *noise*.

Many NDR methods have been developed over the last two decades due to the lack of accuracy and applicability of classic Linear Dimensionality Reduction (LDR) methods such as Principal Component Analysis (PCA) [12], which finds directions of maximum variance, or Multi-Dimensional Scaling (MDS) [13], which attempts to preserve the squared Euclidean distance between pairs of points. As the Euclidean distance used in MDS to quantify the distance between points in the high-dimensional space rather than the actual distance on the manifold, MDS has difficulties of inferring a faithful low-dimensional embedding of non-linear data. The NDR method Isometric Mapping (Isomap), replaces the Euclidean metric in MDS with *geodesic metric* to represent pairwise distances between points, successfully resolves the aforesaid problem in MDS [14]. Although Isomap has been used to analyze low-dimensional embedding of data from several domains, such as collective motion [15], face recognition [16], and hand-writing digit classification [17], this method can suffer from short-circuiting [18], low-density of the data [19], and non-convexity [20], all of which can be magnified in the presence of noise. It is therefore our goal here to propose a new method which ameliorates some of these issues as compared to Isomap.

Generally, NDR approaches reveal smooth low-dimensional and nonlinear manifold representations of high-dimensional data. While there are many unique capabilities provided by current NDR methods, most of them encounter poor performance in specific instances. In particular, many current NDR methods are not adept at preserving the *smoothness* of the embedded manifold when the

[*] Corresponding author.
E-mail addresses: kdgajamannage@wpi.edu (K. Gajamannage), rcpaffenroth@wpi.edu (R. Paffenroth), ebollt@clarkson.edu (E.M. Bollt).

https://doi.org/10.1016/j.patcog.2018.10.020
0031-3203/© 2018 Elsevier Ltd. All rights reserved.



(a)

(b)

Fig. 1. This figure demonstrates the lack of smoothness of the geodesics generated by Isomap. (a) Three nearest neighbors for each point (blue dots) of a spherical dataset of 300 points are found and joined by line segments (shown in blue) to create a graph structure. The Isomap manifold distance between two arbitrary points $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is estimated as the length of the geodesic (red path), that is defined as the shortest path between two points, computed by using, for example, Floyd's algorithm [23]. (b) However, our approach creates a *smoothing spline*, shown by the black curve, that is fitted through the points in the geodesic as a better approximation of the distances on the smooth manifold than the geodesic distance. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

data is sparse or noisy. Isomap closely mimics the underlying manifold's geometry using a graph structure that it makes using a nearest neighbors search [21,22], over the high-dimensional data. The geodesic between two points on the manifold is defined as the shortest path in this graph between those two points. Geodesics are generally *piecewise linear*, thus, the manifold constructed using geodesics in this method is not actually smooth at each node, as demonstrated in Fig. 1(a). Moreover, given a sufficiently smooth manifold, the Isomap generated geodesic distance will generally be an *over-estimate* of the true manifold distance as demonstrated in Fig. 1(b). Of course, such issues are intensive in the presence of sparse and noisy measurements. Accordingly, herein we propose to replace the piecewise linear Isomap geodesic by a *smoothing spline* as shown by the black curve in Fig. 1(b) and *consider the length of the spline as the estimation of the manifold distance between points*.

There are few NDR methods found in the literature that utilize smoothing splines for embedding like our approach. For example, Local Spline Embedding (LSE) also uses smoothing splines to perform the embedding [24]. This method minimizes the reconstruction error of the objective function and embeds the data using smoothing splines that map local coordinates of the underlying manifold to global coordinates. Specifically, LSE assumes the existence of a smooth low-dimensional underlying manifold and the embedding is based on an eigenvalue decomposition that is used to project the data onto a tangent plane. However, differing from our approach, LSE assumes that the data is noise free and unaffected by anomalies. Another disadvantage of LSE is that it embeds the data into a space where the distances in this space are not faithful to the distances on the manifold. The Principal Manifold Finding Algorithm (PMFA) is another NDR method that also uses cubic smoothing splines to represent the manifold and then quantifies the intrinsic distances of the points on the manifold as lengths of the splines [25]. However, this approach embeds high-dimensional data by reducing the reconstruction error over a two-dimensional space. As this method only performs two-dimensional embeddings, its applicability is limited for problems with large intrinsic dimensionality. As we will demonstrate in Section 4, our proposed method overcomes the limitations of the methods LSE and PMFA.

This paper is structured as follows. In Section 2, we will detail the MDS and Isomap algorithms and describe the evolution of our NDR method from these methods. Section 3 presents our NDR method, Smooth Geodesic Embedding (SGE), that fits geodesics, as in Isomap, by smoothing splines. We analyze the performance of the SGE method in Section 4 versus three NDR methods, Isomap, LSE, and PMFA, using three representative examples: a semi-spherical dataset, images of faces, and images of hand written digits. Finally, we provide discussion and conclusions in Section 5.

2. Multidimensional scaling and Isomap

Here we begin by deriving the mathematical details of the LDR method MDS. Then, we proceed to discuss Isomap which replaces the Euclidean distance in MDS by geodesic distance. Next, we derive our method, SGE, as an extension of Isomap that fits geodesics with smoothing splines.

2.1. Multidimensional scaling

Multidimensional scaling is a classic LDR algorithm that leverages the squared Euclidean distance matrix $\boldsymbol{D}^2=[d_{ij}^2]_{n\times n}$; where $d_{ij}=\|\boldsymbol{y}_i-\boldsymbol{y}_j\|_2$ and n is a number of points in the data. Here, $\boldsymbol{y}_i,\,\boldsymbol{y}_j\in\mathbb{R}^{n\times 1}$, are two points on the high-dimensional dataset $\boldsymbol{Y}=[\boldsymbol{y}_1;\ldots;\boldsymbol{y}_i;\ldots;\boldsymbol{y}_j;\ldots;\boldsymbol{y}_n]$. This method first transforms the squared distance matrix $\boldsymbol{D}^2$ into a Gram matrix $\boldsymbol{S}=[s_{ij}]_{n\times n}$, which is derived by *double-centering* [26] the data using

$$s_{ij}=-\frac{1}{2}\Big[d_{ij}^2-\mu_i(d_{ij}^2)-\mu_j(d_{ij}^2)+\mu_{ij}(d_{ij}^2)\Big].\tag{1}$$

Here, $\mu_i(d_{ij}^2)$ and $\mu_j(d_{ij}^2)$ are the means of the i-th row and j-th column, respectively, of the squared distance matrix, and $\mu_{ij}(d_{ij}^2)$ is the mean of the entire matrix D^2. MDS then computes the Eigenvalue Decomposition (EVD) of $\boldsymbol{S}$ as

$$\boldsymbol{S}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T,\tag{2}$$

where $\boldsymbol{U}$ is a unitary matrix ($\boldsymbol{U}^T\boldsymbol{U}=\boldsymbol{I}$) providing the eigenvectors $\boldsymbol{U}^T$ and a diagonal matrix of eigenvalues $\boldsymbol{\Sigma}$. The Gram matrix $\boldsymbol{S}$, that is made from the squared Euclidean distance matrix $\boldsymbol{D}^2$, is symmetric and Semi-Positive Definite (SPD)[1] [26]. Thus, all the eigenvalues of an SPD matrix $\boldsymbol{S}$, and both the Singular-Value Decomposition (SVD) and the EVD of $\boldsymbol{S}$ are the same [26]. $\boldsymbol{\Sigma}$ and $\boldsymbol{U}^T$ are arranged such that the diagonal of $\boldsymbol{\Sigma}$ contains the eigenvalues of S in descending order, and the columns of $\boldsymbol{U}^T$ represent the corresponding eigenvectors in the same order. We estimate p-dimensional latent variables of the high-dimensional dataset by

$$\hat{\boldsymbol{X}}=\boldsymbol{I}_{p\times n}\boldsymbol{\Sigma}^{1/2}\boldsymbol{U}^T.\tag{3}$$

Here, $\boldsymbol{I}_{p\times n}$ is a matrix made from first p rows of the identity matrix $\boldsymbol{I}_{n\times n}$ and $\hat{\boldsymbol{X}}$ is the p-dimensional embedding of the input data $\boldsymbol{Y}$. However, due to both the approximations in our method and finite precision in computer arithmetic, the computed $\boldsymbol{S}$ might deviate slightly from being SPD. The EVD of such an $\boldsymbol{S}$ might have small negative eigenvalues and these negative eigenvalues would violate Eq. (3). Accordingly, as we will discuss in Section 2.2, we replace the EVD on $\boldsymbol{S}$ by the SVD. Multidimensional scaling has limited applicability as it is inherently a linear method. However, the NDR scheme Isomap overcomes this drawback by employing geodesic distance instead of Euclidean distance.

2.2. Isomap

Isomap creates a graph structure, based upon high-dimensional data, that estimates the intrinsic geometry of the manifold. The graph structure created by Isomap can be parameterized in multiple ways, but herein we focus on the parameter δ which measures the number of *nearest neighbors* to a given point [22]. The

[1] A symmetric $n\times n$ matrix $\boldsymbol{M}$ is said to be SPD, if $\boldsymbol{z}^T\boldsymbol{M}\boldsymbol{z}\geq 0$ for all non-zero $\boldsymbol{z}\in\mathbb{R}^{n\times 1}$.

nearest neighbor collection for each point is transformed into a graph structure by treating points as graph nodes and connecting each pair of nearest neighbors by an edge having the weight equal to the Euclidean distance between the two points. Given such a graph, the distance between any two points is measured as the *shortest path distance in the graph*, which is commonly called the *geodesic distance*.

The geodesic distance between any two points in the data can be computed in many ways, including Dijkstra's algorithm [27], one that the original Isomap used. Dijkstra's algorithm, having computational complexity of $\mathcal{O}(n^2)$ when used for adjacency matrices, computes the shortest path between two pairs of points at a time [28]. Since our dataset has $n(n-1)/2$ distinct pairs of points (we make combinations of 2 points out of n points), the total complexity of the Dijkstra's algorithm is $\mathcal{O}(n^4/2)$, $[\mathcal{O}(n^3(n-1)/2) \approx \mathcal{O}(n^4/2)]$. However, Floyd's algorithm [23] computes shortest paths between all pairs of points in one batch with the computational complexity of $\mathcal{O}(n^3)$ [28], which is more efficient than utilizing Dijkstra's algorithm. Thus, we replace Dijkstra's algorithm in Isomap with Floyd's algorithm.

As in MDS, we first formulate the doubly centered matrix $\boldsymbol{S}$ from the squared geodesic distance matrix using Eq. (1). The doubly centered matrix here is not necessarily SPD as we *approximate* the true geodesic distance matrix by the shortest graph distance [26]. In fact, our computational process uses several numerical approximations that might also cause $\boldsymbol{S}$ to deviate from being SPD. Thus, the eigenvalue decomposition of matrix $\boldsymbol{S}$ might produce negative eigenvalues and Eq. (3) does not hold in this case. To overcome this problem, it is the standard to perform the SVD over the Gram matrix $\boldsymbol{S}$ as

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^T, \tag{4}$$

where Σ is a diagonal matrix of singular values (nonnegative), and U and V are unitary matrices. The p latent variables of the higher dimensional input data are revealed by Eq. (3) with $\boldsymbol{\Sigma}$ and $\boldsymbol{U}^T$ obtained from Eq. (4).

Isomap emphasizes nonlinear features of the manifold. However, the lengths measured using geodesics might not faithfully reflect the true manifold distance, as we demonstrate in Fig. 1. Accordingly, we propose to overcome this drawback in Isomap by utilizing a smoothing approach for geodesics.

3. Smoothing geodesics embedding

Our goal is to fit the geodesics computed in Isomap with smoothing splines to closely mimic the manifold and preserve the geometry of the embedding. Classic smoothing spline constructions [29] require one input parameter, denoted by s, that controls the smoothness of the spline fitted through the points in a geodesic. Our proposed method, SGE, has five parameters:

- δ (inherent from Isomap) controls the number of nearest neighbors,
- μ_s controls the smoothness of the splines,
- ν controls the threshold of the length of splines before reducing the order of the spline to the next level,
- h controls the number of discretizations that the method uses to evaluate the length of a spline, and
- finally, p prescribes the number of embedding dimensions (latent variables).

Note that, we will provide details of the parameters μ_s, ν, and h later in this section.

Here, we demonstrate our approach by fitting a spline over an arbitrary geodesic $\mathcal{G}$, having $m \geq 2$ points, in the graph created by a nearest neighbors search algorithm. For an index k we have that d-dimensional points in $\mathcal{G}$ are given by

$$\left\{\boldsymbol{y}_k = [y_{1k}, \ldots, y_{dk}]^T | k = 1\ldots, m\right\}. \tag{5}$$

For each dimension $l \in \{1, \ldots, d\}$, we fit $\{y_{lk}|k = 1\ldots, m\}$ using one dimensional smoothing splines $\hat{f}_l(z)$ of order $\theta + 1$ that are parameterized in $z \in [0,\ 1]$ by minimizing

$$\sum_{k=1}^{m} \left[y_{lk} - \hat{f}_l(z_k)\right]^2 + s\int_0^1 \left[\hat{f}_l^{(\theta)}(z)\right]^2 dz \tag{6}$$

as in [29]. Here, (θ) represents the order of the derivative of $\hat{f}_l$, and z_k is a discretization of the interval [0,1] such that $z_1 = 0$, $z_k = (k-1)/(m-1)$, and $z_m = 1$. Minimizing of Eq. (6) yields d one-dimensional smoothing splines $\{\hat{f}_l(z)|l = 1, \ldots, d\}$. We combine these one dimensional splines and obtain a d-dimensional smoothing spline of the points $\{\boldsymbol{y}_k|k = 1\ldots, m\}$ in $\mathcal{G}$ as,

$$\hat{\boldsymbol{f}}(z) = [\hat{f}_1(z), \ldots, \hat{f}_d(z)]^T, \tag{7}$$

which is called the *smooth geodesic*. In numerical implementations, the order $\theta + 1$ of the spline $\hat{f}$ should be less than number of points m in the geodesic [29].

Choosing the order of a spline θ is challenging, since while a spline with some specified order might perfectly fits the data, another spline with a different order might weakly fits the data. The length of the fitted spline between two points is defined as the manifold distance between those two points, thus an over-fitted spline might provide an incorrect manifold distance. To overcome this problem, we introduce the spline threshold ν (in percentage) which allows the maximum length of a spline that can yield beyond the length of the corresponding geodesic. If the length of a spline with a specific order exceeds this limit, SGE keeps on reducing the order of the spline by one unit until the length of the new spline satisfies the threshold. If none of the orders satisfy the threshold, then SGE assumes the manifold distance is the default distance which is defined to be the geodesic distance. We opt for this procedure, as it is worthwhile to fit a spline with a lower order when a higher order spline fails numerically. Choosing the order of the smoothing spline can also be considered as a trial and error process. For simplicity, we choose the order here by using the threshold percentage ν. Again for the simplicity, we start by fitting a cubic smoothing spline over the points on a given geodesic and then reduce the order if the length doesn't meet the threshold. Cubic smoothing splines emphasize smoothness while involving a low fitting error. We empirically observed that over-fitting occurs very rarely in SGE, thus most of the geodesics were fitted with cubic smoothing splines.

Below, we present our procedure of choosing the order of a spline, fitting points $\{\boldsymbol{y}_k|k = 1\ldots, m\}$ on a geodesic, under three main cases (1, 2, and 3) and some sub-cases (a, b,...):

- **Case–1** If $m \geq 4$:
  - **Case–a:** we first fit the points in the geodesic with a cubic smoothing spline $\hat{\boldsymbol{f}}(z)$ where $z \in [0,\ 1]$ according to Eqs. (6) and (7). Note that, a cubic smoothing spline is represented by $\theta = 2$ in Eq. (6). We discretize this spline into h segments $z_{k_1} = (k_1 - 1)/(h-1)$; $k_1 = 1, \ldots, h$ and compute the length,

$$d_{\hat{\boldsymbol{f}}} = \sum_{k_1=1}^{h-1} \|\hat{\boldsymbol{f}}(z_{k_1+1}) - \hat{\boldsymbol{f}}(z_{k_1})\|. \tag{8}$$

    Then, the length $d_{\hat{\boldsymbol{f}}}$ is compared with the corresponding geodesic distance

$$d_{\mathcal{G}} = \sum_{k=1}^{m-1} \|\boldsymbol{y}_{k+1} - \boldsymbol{y}_k\|. \tag{9}$$

If $d_{\hat{\boldsymbol{f}}} < d_{\mathcal{G}}(100+\nu)/100$ (so that ν is thought of as a percentage), then we accept $d_{\hat{\boldsymbol{f}}}$ as the length of the smooth geodesic, otherwise we proceed to Case–b. The parameter ν (in percentage) defines the threshold (the upper bound) of the length of the spline $\hat{\boldsymbol{f}}$ that is allowed to exceed from the length of the corresponding geodesic.

- **Case–b:** we fit the data with a quadratic (i.e., $\theta = 1$) spline $\hat{\boldsymbol{f}}$ according to Eqs. (6) and (7) and compute the length of the quadratic spline using Eq. (8). If $d_{\hat{\boldsymbol{f}}} < d_{\mathcal{G}}(100+\nu)/100$, then we accept $d_{\hat{\boldsymbol{f}}}$ as the length of the smooth geodesic, otherwise proceed to Case–c.
- **Case–c:** we make a linear (i.e., $\theta = 0$) fit $\hat{\boldsymbol{f}}$ according to Eqs. (6) and (7), and measure the length using Eq. (8). If $d_{\hat{\boldsymbol{f}}} < d_{\mathcal{G}}(100+\nu)/100$ in the linear fit, then we accept $d_{\hat{\boldsymbol{f}}}$, otherwise proceed to Case–d.
- **Case–d:** instead of fitting a spline, we assume the original geodesic itself as the fit and treat $d_{\mathcal{G}}$ as the length of the smooth geodesic.

- **Case–2** If $m = 3$:
  The spline fitting process here is started from fitting a quadratic spline as only three points are in the geodesic. Thus, we carry-out all the Cases b–d as in Case–1.
- **Case–3** If $m = 2$:
  We have only two points in the geodesic, thus, we perform Cases c–d as in Case–1.

We use the smoothing parameter s to offset the spline fit between no fitting error (when $s = 0$) and the best smoothness (when $s \to \infty$). The parameter s controls the sum of square errors between the training points and the fitted function. The best value for s ensuring the least error while having a sufficient smoothness is bounded by a function of the number of points in the geodesic as

$$m - \sqrt{m} \le s \le m + \sqrt{m}, \tag{10}$$

[30]. Since the number of points in geodesics vary, we are unable to input a one-time value as the smoothing parameter into the method that satisfies the inequality (10). In order to control this, here we introduce a new parameter called the smoothing multiplier $\mu_s \ge 0$ such that $s = \mu_s m$. Thus, for input parameter μ_s, such that

$$1 - 1/\sqrt{m} \le \mu_s \le 1 + 1/\sqrt{m}, \tag{11}$$

SGE uses different smoothing levels for different splines based on number of points on the geodesics (m).

For each pair of point in the dataset, say they are indexed by i and j, we execute the aforesaid procedure and approximate the length of the smooth geodesic d_{ij}. Then, we square the entries d_{ij} and create the matrix $\boldsymbol{D}^2 = [d_{ij}^2]_{n\times n}$. We perform double centering on $\boldsymbol{D}^2$ using Eq. (1) to obtain the doubly centered matrix $\boldsymbol{S}$. Then, we compute SVD as in Eq. (4) followed by computing p-dimensional latent variables $\hat{\boldsymbol{X}}$ according to Eq. (3). A summary of the SGE method is presented in Algorithm 1.

Approximate geodesics arising from graph shortest paths in a finite dataset are different than the true geodesics. However, smoothing splines that fit points on geodesics are capable of closely approximating the true geodesics of finite, sparse, and noisy datasets sampled from a manifold, as shown in Fig. 2.

The smoothing spline approach in SGE approximates true geodesic distance of sparse samples of data more accurately than that of the graph distance used in Isomap [Fig. 2(a)]. Note, the shortest path between two points on a noise free manifold converges to the true geodesic of the manifold as the number of sample points approaches infinity [31]. Thus, Isomap can convergently approximate the manifold distances using shortest graph distances and makes better predictions with dense samples of data. However, as SGE fits vertices on shortest paths with smoothing splines, we demonstrate herein that SGE converges to the true manifold distance at a faster rate than that of Isomap. In particular, our smoothing approach assures better predictions than that of Isomap even under sparse samples of data as we justify in Section 4.

Algorithm 1 Smooth Geodesics Embedding (SGE). Inputs: Data ($\boldsymbol{Y}$), number of nearest neighbors (δ), smoothing multiplier (μ_s), spline threshold percentage (ν), number of discretizations (h), and embedding dimensions (p). Outputs: List of p largest singular values (λ_l; $l = 1, \ldots, p$) and p-dimensional embedding ($\hat{\boldsymbol{X}}$).

```
1: For each point in $\boldsymbol{Y}$, choose $\delta$ number of nearest points as neighbors [21].
2: Consider all the point in $\boldsymbol{Y}$ as nodes and if any two nodes are chosen to be neighbors in 1, then join them by an edge having the length equal to the Euclidean distance between them. This step converts the dataset into a graph.
3: For each pair of nodes in the graph, find the points $\mathcal{G} = \{\boldsymbol{y}_k | k = 1\ldots, m\}$ in the shortest path using the Floyd's algorithm [23]. Here, $m = |\mathcal{G}| \ge 2$.
4: The points in $\mathcal{G}$ are fitted with a smoothing spline and its length is computed:
   Case–1 ($m \ge 4$):
   Case–a:
      Fit $\mathcal{G}$ with a cubic smoothing spline using Eqs. (6) and (7), then approximate the length $d_{\hat{\boldsymbol{f}}}$ of the spline using Eq. (8). Let, the length of the geodesic is $d_{\mathcal{G}}$ [Eq. (9)]. If $d_{\hat{\boldsymbol{f}}} < d_{\mathcal{G}}(100+\nu)/100$, then accept $d_{\hat{\boldsymbol{f}}}$ as the length of the smooth geodesic, otherwise proceed to Case–b.
   Case–b:
      Fit $\mathcal{G}$ with a quadratic smoothing spline using Eqs. (6) and (7). Approximate the length $d_{\hat{\boldsymbol{f}}}$ of the spline using Eq. (8). If $d_{\hat{\boldsymbol{f}}} < d_{\mathcal{G}}(100+\nu)/100$, then accept $d_{\hat{\boldsymbol{f}}}$ as the length of the smooth geodesic, otherwise proceed to Case–c.
   Case–c:
      Fit $\mathcal{G}$ with a linear smoothing spline using Eqs. (6) and (7). Approximate the length $d_{\hat{\boldsymbol{f}}}$ of the spline using Eq. (8). If $d_{\hat{\boldsymbol{f}}} < d_{\mathcal{G}}(100+\nu)/100$, then accept $d_{\hat{\boldsymbol{f}}}$ as the length of the smooth geodesic, otherwise proceed to Case–d.
   Case–d:
      Consider $d_{\mathcal{G}}$ as the approximated length of the smooth geodesic.
   Case–2 ($m = 3$): Perform Cases b–d similarly as in Case–1.
   Case–3 ($m = 2$): Perform Cases c–d similarly as in Case–1.
5: Fill the distance matrix $\boldsymbol{D}^2 = [d_{ij}^2]_{n\times n}$ where $d_{ij}$ is the length of the smooth geodesic between nodes $i$ and $j$ computed in 3–4. Double center $\boldsymbol{D}^2$ and convert it to a Gramian matrix $\boldsymbol{S}$ using the Eq. (1).
6: Perform the SVD on $\boldsymbol{S}$ using Eq. (4) and extract $p$ largest singular values $\lambda_l$; $l = 1, \ldots, p$ along with the latent variable $\hat{\boldsymbol{X}}$ as given by Eq. (3).
```

However, both the smoothing spline approximation of noisy data in SGE and the geodesic approximation of noisy data in Isomap, might not faithfully represent the real manifold [Fig. 2(b)]. This is because the Floyd's algorithm might find a shortest path that is different than the true manifold if the data is contaminated with noise. Since smoothing splines fit the data points on such shortest paths, they might also deviated from the true man-

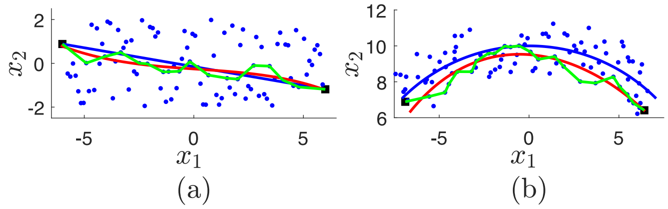


Fig. 2. Trade-offs of the shortest paths and smoothing splines from the true geodesics. (a) The first dataset (blue points) which is noise free is sampled from a two-dimensional rectangular shaped manifold. (b) The second dataset (blue points) is sampled from a one-dimensional manifold (blue curve) representing an arc after imposing a uniform noise. We run nearest neighbor search algorithm over both datasets with four nearest neighbors ($\delta = 4$) and create a graph structure in each dataset. Then, we compute the geodesics (green curves) between two points (black squares) in the datasets, and then fit the points on each geodesic using a cubic smoothing splines (red curves) with $\mu_s = 1$. Note that, the blue curves represent the true manifold distance between two black squares. In (a) we see that the smoothing spline more faithfully representing the true geodesic distance of on the noise free manifold, while in (b) we see that both SGE and Isomap can suffer in the presence of noise. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

ifold. Thus, both geodesics in Isomap and smoothing splines in SGE demonstrate lack of convergence to the true manifold even at the limit of infinite samples of noisy data. We believe that we can produce a convergent version of the SGE method, if we first compute the curvature of the manifold and then utilize a technique that helps to choose the shortest paths close to the manifold. We will explain this technique with details as a future work in Section 5. However, we demonstrate that, empirically speaking, the smoothing spline approach is a better replacement for graph shortest paths even when the data is contaminated with noise using the examples in Section 4.

Note that, we provide more examples in Section 4 that support the accuracy of embedding noisy datasets versus that of sparse datasets. The reason for that is, convergence of both methods, Isomap and SGE, of embedding sparse datasets is supported by [31], while the convergence of both methods is not guaranteed when the data is contaminated with noise.

4. Performance analysis

In this section, we demonstrate the effectiveness SGE, versus Isomap and PMFA, in example 1 and then that of SGE versus Isomap in examples 2 and 3. PMFA first makes n_c^1 non-overlapping slices of data along the direction of the largest singular vector and then makes n_c^2 slices along the second largest singular vector. Then, the data in each slice is fitted with a cubic smoothing spline of smoothing parameter s_r. The user input parameters in this method are n_c^1, n_c^2, and s_r. The grid structure represented by all the cubic smoothing splines is used as the local intrinsic coordinate system that we use to measure the embedding distances. Large n_c^1 and n_c^2 values make thin slices with few points. Accordingly, the low density of points in such a slice can cause cubic smoothing splines to weakly fit the data and misinterpret the manifold. In contrast, small n_c^1 and n_c^2 make few slices and create a sparse grid structure. This sparse grid structure might loose the geometry of the manifold. Accordingly, we use moderate values for the parameters n_c^1 and n_c^2, say 10 each, in the examples we provide. While big values of s_r make less oscillatory cubic smoothing splines, small values can make highly oscillatory cubic smoothing splines. As stated in [25], the best value for s_r is 0.9, and we use that value in the numerical examples in this manuscript.

LSE is also an NDR method that employs a spline approach in their embedding. This method requires one input parameter for a number of nearest neighbors (δ) and it projects δ neighbors of each point into a tangent space with local coordinates. Each such local coordinate is then mapped to its own single global coordinate with respect to the underlying manifold using splines. The parameter δ in LSE is also an input parameter in SGE, thus, details for choosing the value for this parameter will be explained later when the parameter values of SGE are explained.

For all the examples in this section, we set $\nu = 10\%$ and $h = 100$ in SGE. Setting ν to a high value increases the tendency of fitting points on the geodesics in SGE with cubic smoothing splines than fitting those points with splines having order less than three. SGE is fabricated to reveal a smooth underlying manifold that is ensured by cubic smoothing splines than a spline with a low order. However, high ν values sometimes over-fit the data and that will then yield inaccurate embedding. We empirically learned that setting ν to 10% can exclude both of aforesaid extremes. Each spline is discretized to h segments and the length of the spline is computed as sum of linear lengths of these segments. While a big h gives a very accurate length for the spline, it increases the computational time as SGE has to compute lengths of $n(n-1)/2$ splines for a dataset of n points. Thus, we set $h = 100$ since the accuracy of the spline lengths by 100 discretization is satisfactory for our study.

We set $\delta = 3$ or 4, and $\mu_s = 1$ in SGE, if not stated otherwise. Each point in the dataset is adjacent to δ number of nearest neighbors and the graph structure is made. Setting a big value for δ will create more edges in the graph and that might loose the topology of the graph as geodesics might not infer the true curvature of the manifold in this case. However, a small value of δ might produce multiple connected components in the graph where SGE treats the large connected component and neglects others in this case. We set $\delta = 3$ or 4, since we empirically observed that these values stay in the middle of aforesaid extremes. Choosing a best value for μ_s is challenging, thus based on Eq. (11), we set $\mu_s = 1$, since this value provides both a perfect smoothness and a better fit for the splines.

The NDR methods that we utilize in this section should preserve the pairwise distances between data and embedding in order to compare\contrast them using two distance metrics [Eqs. (14) and (17)] that we use to compute the embedding error in this paper. To visualize an instance of embedding of these four methods SGE, Isomap, PMFA, and LSE, we embed a dataset sampled from a semi-sphere of 600 points [Fig. 3(a)] defined by

$$\begin{aligned} y_1 &= r\cos(\gamma_1)\cos(\gamma_2), \\ y_2 &= r\cos(\gamma_1)\sin(\gamma_2), \\ y_3 &= r\sin(\gamma_1), \end{aligned} \tag{12}$$

for $\gamma_1 = \mathcal{U}[-\pi/2, \pi/2]$ and $\gamma_2 = \mathcal{U}[0, \pi]$, where $\mathcal{U}[a, b]$ denotes a uniform distribution between a and b. Here, r is the radius of the semi-sphere which is set to $20 + \mathcal{N}[0, \eta^2]$, where $\mathcal{N}[0, \eta^2]$ is a random variable sampled from a Gaussian distribution with mean 0 and variance η^2. We set $\eta = 0$ as we need this semi-sphere to be noise free.

We compute two dimensional embedding of this semi-sphere [Fig. 3(b–e)] using Isomap with $\delta = 3$; LSE with $\delta = 3$; PMFA with $n_c^1 = n_c^2 = 10$ and $s_r = 0.9$; and SGE with $\delta = 3$, $\mu_s = 1$, $\nu = 10\%$, and $h = 100$. According to Fig. 3, moving from data to embedding, LSE shrinks the distances in the embedding while others seem preserve the original distance between points. Thus, we omit LSE from this analysis and only rely on the rest of the methods since two distance preserving error metrics that we used here can't be implemented for LSE. Moreover, PMFA is computationally expensive when the data is high dimensional (as stated in [25]) like in face images of example 2 where the dataset is 4096 dimensions, and hand written digits of example 3 where the dataset is 784 dimen-

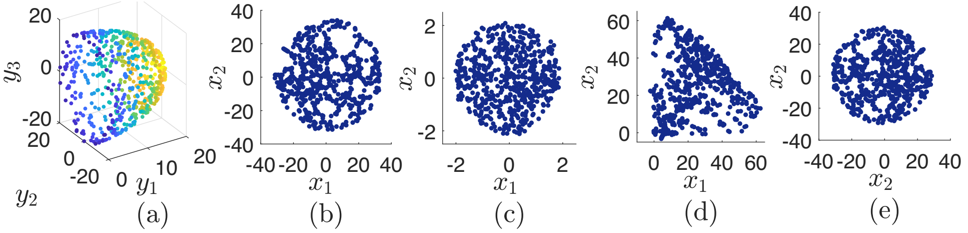


Fig. 3. Two dimensional embedding of (a) a noise free semi-sphere of 600 points using (b) Isomap, (c) LSE, (d) PMFA, and (e) SGE.

sions. Thus, we omit the implementation of PMFA for the datasets in those two examples.

As the first example, we use a synthetic three dimensional dataset of a semi-sphere to analyze the performance of SGE with respect to neighborhood size (δ), smoothness (μ_s), sparsity (n), and noise (η). Then, we study the performance of SGE using two high-dimensional standard benchmark datasets: 1) face images [32]; and 2) images of handwritten digits (2's, 4's, 6's, and 8's) [33]. We analyze the performance of SGE versus Isomap and PMFA in example 1, and that of SGE versus Isomap in examples 2 and 3.

4.1. Embedding of a semi-sphere

We begin this section by embedding a synthetic dataset, sampled from a semi-spherical manifold, using SGE, Isomap, and PMFA to demonstrate the key concepts of our proposed SGE technique since, in this case, we can analytically compute the manifold distance on the semi-sphere and then compare it with the embedding distances computed by above NDR methods.

First, we compare the performance of SGE with changing δ and μ_s by embedding a sample 600 points generated from the manifold defined by Eq. (12) with $\eta = 2$. Both μ_s in SGE and s_r in PMFA are parameters to control the smoothness of the splines, however, while μ_s can only take any nonnegative value, chosen s_r should be in [0, 1] [25]. Since we are unable to compare SGE and PMFA in this context, we only provide here a comparison between Isomap and SGE.

We set the spline threshold ν and spline discretization h to be 10% and 100, respectively, and run the SGE algorithm repeatedly over the spherical dataset with $\delta = 2, 3, \ldots, 8$; $\mu_s = 0, 0.1, \ldots, 1.0$ and obtain two-dimensional embeddings. Here, we have 77 different pairs of δ's and μ_s's, those then produce 77 two-dimensional embeddings. Now, we asses the performance of the methods in terms of distance preserving ability between the original data and the embedding. For each such embedding (77 in total), we compute distances between points in the embedding space using the Euclidean distance metric that we denote by $\boldsymbol{D}_S$. Now, we run Isomap with the same sequence of δ's above and obtain its two-dimensional embeddings. The Euclidean distance matrix for the embedding of Isomap is denoted by $\boldsymbol{D}_I$. Now, we compute the true manifold distances between points of the dataset using the cosine law. If $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are two points on a semi-sphere with radius r, the manifold distance d is given by

$$d = r\gamma; \;\; \gamma = \cos^{-1}\left(\frac{\boldsymbol{\alpha}\boldsymbol{\beta}}{|\boldsymbol{\alpha}||\boldsymbol{\beta}|} \right), \tag{13}$$

[34]. We compute all the pairwise distances using Eq. (13) and form the distance matrix $\boldsymbol{D}_M$ of the data sampled on the manifold.

The embedding error of SGE, denoted by $\mathcal{E}_S$, is computed as the Mean Absolute Deviation (MAD) between embedding and data [35]. Since the distance matrices are symmetric and have zeros on the diagonal, MAD can then be computed using

$$\mathcal{E}_S = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left| (\boldsymbol{D}_M)_{ij} - (\boldsymbol{D}_S)_{ij} \right|. \tag{14}$$

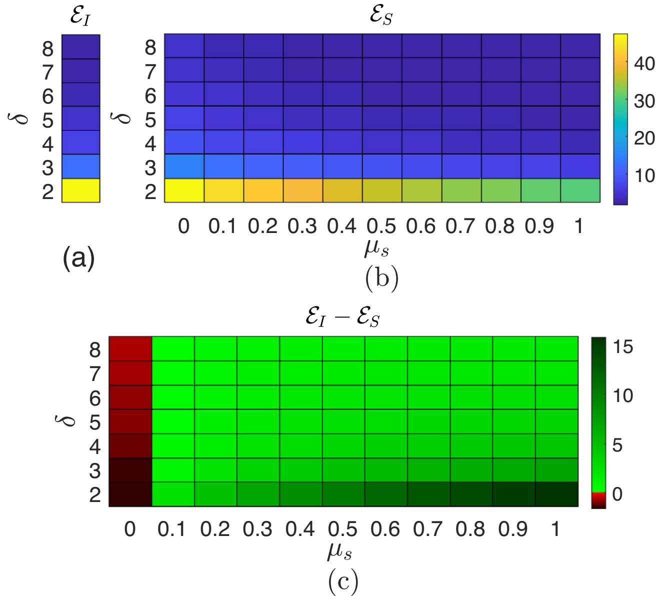


Fig. 4. Analyzing the performance of Isomap and SGE embeddings using Mean Absolute Deviation (MAD). Herein, we compute, (a) MAD between Isomap embedding and data, denoted by $\mathcal{E}_I$, for different neighborhood sizes (δ's), and (b) MAD between SGE embedding and data, dented by $\mathcal{E}_S$, for different neighborhood sizes and smoothing multiplier (μ's). (c) The difference of errors between these two methods ($\mathcal{E}_I - \mathcal{E}_S$) is computed in the variable space δ and μ_s. *The green cells denote that the performance of SGE is superior to that of Isomap*. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Similarly, for each pair of δ and μ_s, we also compute MAD between the embedding of Isomap and the original data that we denote by $\mathcal{E}_I$. Fig. 4 illustrates MADs for Isomap ($\mathcal{E}_I$), MADs for SGE ($\mathcal{E}_S$), and their differences ($\mathcal{E}_I - \mathcal{E}_S$), versus δ and μ_s. Fig. 4(a) and (b) show that both methods display decreasing errors for increasing δ's (i.e., increasing neighbors). Moreover, SGE has a decreasing error as μ_s increases. Fig. 4(c) also indicates that SGE performs better than Isomap for larger smoothing multipliers *for all δ's*. This plot also shows that SGE performs worst when $\delta = 2$ and $\mu_s = 0$, and performs best when $\delta = 2$ and $\mu_s = 1$, as compared to isomap.

Next, we analyze the influence of data sparsity on the manifold for embedding with SGE and compare this to PMFA and Isomap. For that task, we produce a sequence of spherical datasets with an increasing number of points. We create the first dataset with 200 points using Eq. (12) with $r = 20 + \mathcal{N}[0, 2^2]$, then add another 100 points, generated using the same equation, into the first dataset to produce the second dataset. Similarly, we generate the last dataset of 1200 points. Then, we embed these datasets in two-dimensions using Isomap with $\delta = 3$; PMFA with $n_c^1 = n_c^2 = 10$ and $s_r = 0.9$; and SGE with $\delta = 3$, $\mu_s = 1$, $\nu = 10\%$, and $h = 100$. We compute the embedding errors $\mathcal{E}_I$, $\mathcal{E}_P$, and $\mathcal{E}_S$ using MAD for each dataset as explained before. Since a significantly high noise is used for the datasets, we create 16 such sequences of datasets and perform

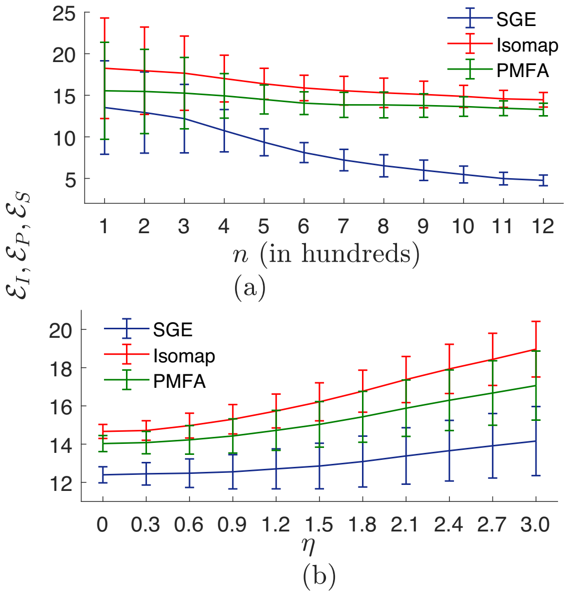


Fig. 5. Mean embedding error of Isomap (in red, denoted by $\mathcal{E}_S$), PMFA (in green, denoted by $\mathcal{E}_P$), and SGE (in blue, denoted by $\mathcal{E}_S$), versus, (a) sparcity and (b) noise. Error bars represent standard deviations of errors computed over realizations. *Note that, SGE has lower average error than that of Isomap and PMFA.* (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

this analysis for16 realizations to allow us to compute averages. Fig. 5(a) shows the mean of embedding errors of 16 realizations and error bars for SGE, Isomap, and PMFA. We observe that the error associated with SGE embedding is smaller than that of Isomap and PMFA for *all the values of n*. This observation demonstrates the advantages of dealing with sparse data when comparing SGE to Isomap and PMFA.

Finally, we study the embedding errors of those three methods in terms of the size of the noise present in the data. For that task, we create a latticed semi-sphere of 600 points using Eq. (12) with appropriately discretized $\gamma_1 \in [-\pi/2, \pi/2]$ and $\gamma_2 \in [0, \pi]$. Then, we impose increasing uniform noise levels into the parameter representing the radius as $r = 20 + \eta \mathrm{U}[-1, 1]$; $\eta = 0, 0.3, 0.9, \ldots, 3$ and produce a sequence of 11 datasets. We embed each dataset using Isomap with $\delta = 3$; using PMFA with $n_c^1 = n_c^2 = 10$ and $s_r = 0.9$; and using SGE with $\delta = 3$, $\nu = 10\%$, $\mu_s = 1$, and $h = 100$. We create 25 such sequences of datasets and perform this analysis for 25 realizations to allow us to compute averages. Fig. 5(b) presents mean embedding errors and error bars for all three methods computed using Eq. (14). We observe that, while $\mathcal{E}_S$ slowly increases with increasing η, $\mathcal{E}_I$ and $\mathcal{E}_P$ increase quickly with increasing η.

4.2. Embedding of face images

In this section, we validate the SGE method using a real-world dataset of face images available in [32]. This dataset consists 698 images each of 64 × 64 dimension with a varying pose and direction of lighting, as shown by a sample of 16 snapshots in Fig. 6(a). We randomly choose 400 images as our baseline dataset and generate three other datasets of 400 images from the baseline dataset by imposing Gaussian noise with standard deviations (σ's) 0.1, 0.2, and 0.3 [Fig. 6(b)]. We set $\delta = 4$, $\nu = 10\%$, $h = 100$ in SGE and run this algorithm over each dataset (4 in total) 5 times for μ_s =0, 0.3, 0.6, 0.9, 1.2. Then, we embed these four datasets in two dimensions using Isomap and LSE with $\delta = 4$.

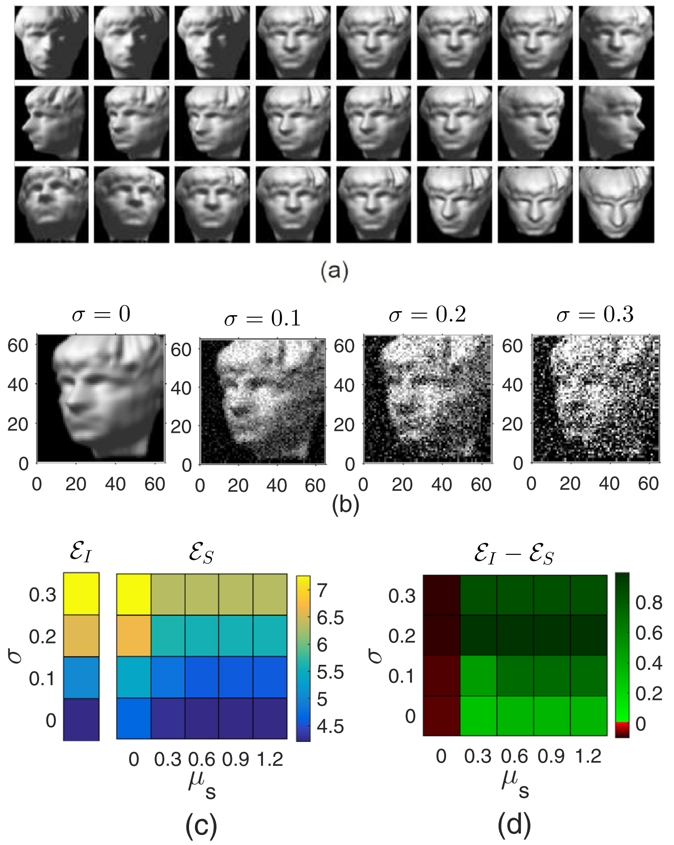


Fig. 6. Embedding of face images (64 × 64 dimensional), distorted with different noise levels, using Isomap and SGE with different smoothing levels. (a) A sample of 16 face images [32]; where the snapshots in the first, second, and third rows, represent left-right light variation, left-right pose variation, and up-down pose variation, respectively. (b) Face images are distorted by imposing three levels of Gaussian noise σ = 0.1, 0.2, and 0.3. The datasets (four in total) are embedded using Isomap and then using SGE with smoothing multipliers μ_s = 0, 0.3 0.6, 0.9, 1.2. Then, (c) the embedding errors of Isomap ($\mathcal{E}_I$) and SGE ($\mathcal{E}_S$), and (c) their error difference are computed. *The green cells denote that the performance of SGE is superior to that of Isomap.* (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

We use the ability to preserve distances between the original and the embedding data to analyze the performance of the method [36]. In particular, we consider the distances in the original (noise free) imagery as the “true” distances and judge the algorithm's ability to recover those distances after the imagery has been corrupted by noise. For both the data and the embedding, we first search δ nearest neighbors for each point and then produce a weighted graph by treating points in the dataset as nodes and connecting each two neighbors by an edge having the length equal to their Euclidean distance. The weighted graph constructed through the nearest neighbor search is a simple graph[2] that does not contain self-loops or multiple edges. We compute the *ij*-th entry of the adjacency distance matrix A for the data as

$$A_{ij} = \begin{cases} d(i,j) & : \text{if } \exists \text{ an edge } ij \text{ in the graph} \\ & \quad \text{of the original data,} \\ 0 & : \text{otherwise,} \end{cases} \tag{15}$$

and the *ij*-th entry of the adjacency distance matrix $\tilde{A}$ for the embedding data as

$$\tilde{A}_{ij} = \begin{cases} d(i,j) & : \text{if } \exists \text{ an edge } ij \text{ in the graph} \\ & \quad \text{of the embedding data,} \\ 0 & : \text{otherwise.} \end{cases} \tag{16}$$

[2] A simple graph is an undirected graph that does not contain loops (edges connected at both ends to the same vertex) and multiple edges (more than one edge between any two different vertices) [37].

Here, $d(i, j)$ is the Euclidean distance between nodes i and j. In this paper, we impose Gaussian noise into real-world datasets such as face images and images of handwritten digits (Section 4.3). *Thus, we think of our original data as the uncorrupted data before we impose the noise.*

For n points in the dataset, the error associated with the neighbors' distance is computed as the average of absolute differences between entries of the adjacency distance matrices,

$$\mathcal{E} = \frac{1}{n(n-1)} \sum_{i,j=1}^{n} \left| A_{ij} - \tilde{A}_{ij} \right|, \tag{17}$$

where δ is the neighbor parameter [36].

Fig. 6(c) illustrates the embedding errors of Isomap, denoted by $\mathcal{E}_I$, and embedding errors of SGE, denoted by $\mathcal{E}_S$, for $\sigma = 0$, 0.1, 0.2, 0.3 and $\mu_s =$ 0, 0.3, 0.6, 0.9, 1.2. We observe that the error increases in both methods when the noise in the data increases. However, the error of embedding noisy data can be reduced significantly by choosing an appropriate smoothing multiplier in SGE as shown here. Fig. 6(d) showing the difference of errors ($\mathcal{E}_I - \mathcal{E}_S$) demonstrates that SGE performs better in terms of error than Isomap for all the noise levels with $\mu_s \geq 0.3$.

4.3. Embedding of handwritten digits

Next, we embed handwritten digits available from the Mixed National Institute of Standards and Technology (MNIST) database [33] using SGE and study the performance of the method. This dataset contains 60,000 images of handwritten digits from 0 to 9 each of 28×28 dimensions. We sample two arbitrary datasets for our study, each with 400 images, such that one dataset has only the digit 2 and the other dataset has the digits 2, 4, 6, and 8.

We run Isomap over the dataset having the digit 2 with $\delta = 4$. We run SGE over this dataset two times: first with $\delta = 4$, $\mu_s = 0$, $\nu = 10\%$, and $h = 100$; and second with $\delta = 4$, $\mu_s = 0.6$, $\nu = 10\%$, and $h = 100$. Thus, aforesaid procedure yields three two-dimensional embeddings. We formulate the adjacency distance matrices for the data and embedding using Eqs. (15) and (16), respectively, and compute the error of embedding using Eq. (17). Then, we distort this dataset with a Gaussian noise having $\sigma = 0.2$ and run Isomap with $\delta = 4$. We run the noisy dataset two times in SGE: first with parameters $\delta = 4$, $\mu_s = 0$, $\nu = 10\%$, and $h = 100$; and second with $\delta = 4$, $\mu_s = 0.6$, $\nu = 10\%$, and $h = 100$. The embedding errors for Isomap, SGE with $\mu_s = 0$, and SGE with $\mu_s = 0.6$, are given in Table 1(a). We see in this table that, regardless of the noise present in the data, the error associated with SGE without smoothing is greater than that of Isomap, while that of SGE with smoothing is smaller than that of Isomap. Moreover, the error of embedding is increased when moving from the noisy dataset to the noise free dataset by 0.87 for Isomap, while that is only increased by 0.24 for SGE with $\mu_s = 0.6$. This is due to the fact that setting the smoothing multiplier μ_s to 0.6 allows SGE to recover the manifold corrupted by noisy measurements.

Next, we embed a sample of 400 digits, consisting of 2's, 4's, 6's, and 8's, into two dimensions using Isomap and SGE. We run Isomap over this dataset with $\delta = 4$. Then, run SGE two times: first with $\delta = 4$, $\nu = 10\%$, $\mu_s = 0$, and $h = 100$; and second with $\delta = 4$, $\nu = 10\%$, $\mu_s = 0.9$, and $h = 100$. Thereafter, we distort the dataset with a Gaussian noise having $\sigma = 0.3$ and then run Isomap with $\delta = 4$ followed by running SGE with the same two parameter sets that we used before. Then, we compute the Isomap and SGE errors associated with embedding of noise free and noisy datasets using Eq. (17) that we present in Table 1(b). Similarly to the embedding of the digit 2, regardless of the error in the data, here we also note that the embedding error of SGE *with no smoothing* is greater than that of Isomap, while the embedding error of SGE *with smoothing* is smaller than that of Isomap. Moreover, moving from embedding of noise free data to embedding of noisy data, while the error associated with Isomap is increased by 0.88, that of SGE with $\mu_s = 0.9$ is increased only by 0.21.

Finally, we compare the classification ability of both methods in the presence of high noise. In this example, we define classification as spatially clustering of similar digits. To visualize the classification ability, we construct two-dimensional Isomap and SGE embeddings of the noisy dataset ($\sigma = 0.3$) of digits 2, 4, 6, and 8, that we present in Fig. 7. Therein, we observe that, while Isomap's embedding is unable to maintain clear boundaries between clusters of the same digit, SGE could at least separate numbers 2 and 8 from the rest of the digits. Thus, we can conclude that while Isomap is unable to achieve a clear classification of digits, SGE with $\mu_s = 0.9$ achieves qualitatively better classification, even under the high noise present in the data.

Table 1
Errors of Isomap and SGE embeddings of, (a) a sample of 400 handwritten2's; and (b) a sample of 400 handwritten digits having number 2's, 4's, 6's, and 8's. The first row of (a) shows the error when the dataset of digit 2 is embedded using both Isomap, and SGE with two smoothing coefficients $\mu_s = 0$ and $\mu_s = 0.6$. Then, the dataset is imposed with a Gaussian noise of $\sigma = 0.2$ and embedded using both Isomap, and SGE with $\mu_s = 0$ and $\mu_s = 0.6$ that you see in the second row of (a). The first row of (b) represents the errors of Isomap embedding, and SGE embeddings with $\mu_s = 0$ and $\mu_s = 0.9$, of the noise free version of the sample of digits having the numbers 2, 4, 6, and 8. The second row of (b) represents the errors of Isomap embedding, and SGE embeddings with $\mu_s = 0$ and $\mu_s = 0.9$, of the noisy version of the dataset created by imposing a Gaussian noise with $\sigma = 0.3$.

(a)	Noise	Isomap	SGE	
			$\mu_s = 0$	$\mu_s = 0.6$
Digit "2"	$\sigma = 0$	7.02	7.13	5.86
	$\sigma = 0.2$	7.89	8.19	6.10

(b)	Noise	Isomap	SGE	
			$\mu_s = 0$	$\mu_s = 0.9$
Digits "2", "4", "6", and "8"	$\sigma = 0$	7.38	7.43	6.09
	$\sigma = 0.3$	8.20	8.36	6.30

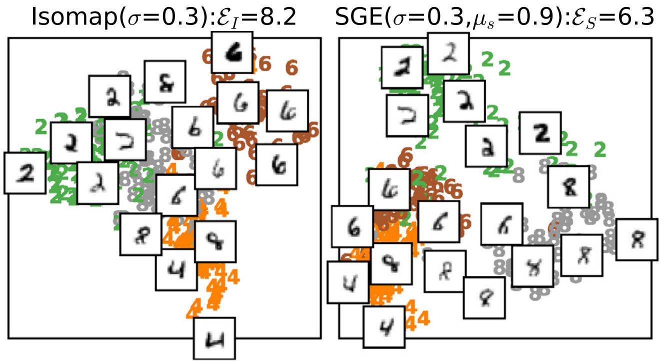


Fig. 7. Isomap and SGE embeddings of handwritten numbers 2, 4, 6, and 8. Different digits are shown in different colors (2's in green, 4's in orange, 6's in brown, and 8's in gray) in the two-dimensional embedding space and the embedded snapshots illustrate the appearance of arbitrarily chosen handwritten digits. The left panel shows the Isomap embedding of the noisy dataset ($\sigma = 0.3$) and the right panel shows the SGE embedding of the same dataset with $\mu_s = 0.9$. The embedding error of each case is indicated in the title of the corresponding panel. Note that, qualitatively speaking, the SGE embedding appears to have better clustering of similar digits than that of Isomap. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5. Conclusion

Nonlinear dimensionality reduction methods can recover unfaithful embeddings due to sparsity and presence of high noise in the data. In order to obtain a faithful embedding for sparse and noisy data, some smoothing procedure should be performed in the embedding. With this idea in mind, herein we introduced a novel nonlinear dimensionality reduction framework using smooth geodesics that emphasizes the underlying smoothness of the manifold. Our method begins by first searching for nearest neighbors for each point using a δ-nearest neighbor search [21]. Then, we create a weighted graph by representing all of the data points as nodes and joining neighboring nodes with edges having their Euclidean distances as weights. For each pair of nodes in the graph, we create a geodesic [14], that is defined as the shortest path between the given nodes, generated using Floyd's algorithm [38]. We replaced Dijkstra's algorithm in classic Isomap with Floyd's algorithm since Floyd's algorithm is more computationally efficient than Dijkstra's algorithm in our case. We fit each such geodesic with a smoothing spline (called a smooth geodesic) that is controlled by two parameters: smoothing multiplier (μ_s) and spline threshold (ν) [29,30]. The length of these splines are treated as manifold distances between corresponding points.

We use a classic MDS method to find the dimension of the distance matrix of smooth geodesics and perform the embedding. The MDS method converts the squared distance matrix to a Gram matrix, employs EVD to compute eigenvalues and eigenvectors, and finally uses Eq. (3) to produce the embedding [26]. In theory, a Gram matrix of a squared distance matrix should be SPD [26]. However, due to geodesic approximation of the true manifold distance and other numerical approximations, this Gram matrix might deviate slightly from being SPD. Since this slightly corrupted Gram matrix produces small negative eigenvalues and those then violate Eq. (3), we replace EVD in MDS by SVD. Note that, the EVD and SVD of an exact Gram matrix are the same.

In SGE, the order of the spline fit is set to either three, two, or one, depending on the spline threshold. Since a sufficient smoothness and a low fitting error can be obtained by cubic smoothing splines, we first rely on a spline fit of order three. However, we observed that the smoothing spline routing in [29] fits very long cubic splines for some specific smoothing multipliers. Thus, if the length of a cubic smoothing spline doesn't satisfy the threshold, we reduce the order of the spline to a lower level. Choosing the order of the spline can also be considered as a trial and error process that we implement here by utilizing a threshold. In future, we will consider replacing this threshold by a trial and error process to obtain a faithful embedding. Hereby, we also be able to consider higher orders for the spline in Eq. (6) than the current highest order of three.

The smoothing spline approach in SGE approximates the true manifold more accurately in many cases than the geodesic approach in Isomap (Fig. 2). Specifically, the use of smoothing splines can closely approximates the true manifold distance even when the data is sparse, in contrast to Isoamp which creates polygonal paths that then add extra length to the true manifold distance. In the limit of infinite number of sample points, the Isomap geodesics converge to the true manifold distance [31]. However, due to the smoothing spline approach in SGE, we have observed that the embedding error of SGE is significantly lower than that of Isomap in many practical problems. This was evidenced by the semi-sphere example in Section 4.1. Although, both methods do not converge to the true manifold when the data is corrupted by noise, SGE emphasizes the smoothness of the manifold while Isomap is severely impacted as the errors in the lengths of the Isomap polygonal paths is intensified.

In the future, we plan to modify the SGE method such that it converges even when the data is corrupted by noise. For that, first, we will need to estimate, or be provided with, the curvature of the manifold described by noisy data. Then, we plan to replace the nearest neighbor search with a range search [22] that finds all the points within a given distance based on the curvature. We will run the range search over the points those are close to the manifold (we can find these points as we know the curvature), and then create a graph by treating data points as nodes and connecting neighbors by edges. We expect that the range search will ensure that the graph doesn't have long edges arising from highly noisy points in data. As the range search is ran over the data points close to the manifold, we expect to be able to create shortest paths those are close to the manifold. Finally, we will follow all the other steps 3–6 as in Algorithm 1 of SGE and compute the embedding.

We first demonstrated the effectiveness of our NDR method on a synthetic dataset representing a semi-sphere. We observed that the smoothing approach provides a better embedding performance than the embedding achieved by standard Isomap or PMFA when embedding a noisy dataset. We also observed that the errors in Isomap and SGE decrease as the neighborhood size increases [Fig. 4(a) and (b)]. However, when the neighborhood size is small, say $\delta = 2,$ SGE has clear performance advantages over Isomap for noisy data when a sufficient smoothness is employed [Fig. 4(c)]. The spherical dataset also demonstrated that SGE is more robust to sparse sampling than Isomap and PMFA [Fig. 5(a)]. Moreover, while increasing noise in the data always appears to reduce the performance of the embedding, irrespective to the method that is used, we see that Isomap and PMFA are highly effected by increasing noise while SGE, with a judicious choice of smoothing multiplier, is more robust [Fig. 5(b)].

We also studied two standard benchmark datasets, face images [32] and handwritten digit images [33], and found that SGE provided similar superior performance on noisy versions of those datasets. In particular, for the digit classification task, we observed that SGE provides qualitatively superior classification performance (that is, clustering similar digits into one group) in the presence of noise. As future work, we will quantify the classification performance of the low-dimensional nonlinear embedding using a variety of standard supervised machine learning techniques.

As a potential application of SGE, we can modify some semi-supervised learning methods such as [39] and metric learnings such as [40] by replacing their LDR routing, applied in unlabeled portions of their data, by SGE. By doing this, we believe that the results of those methods can be improved when they are applied to nonlinear data due to the nonlinear and smoothing features of SGE. Specifically, the similarity evaluation framework in [40] inputs both supervised and unsupervised data. The data here is a collection of patients' claim records, lab test records, and pharmacy records. A distance measure between each pair of patients is generated as the sum of the distances in the supervised data and the unsupervised data. The distances in unsupervised data are computed on the embedding using LDR methods, such as Local Linear Embedding (LLE) [41] and Laplacian eigenmaps [42], might not emphasis nonlinear features of the data. Thus, replacing these LDR method by SGE can produce accurate results in large array of datasets. On the other hand, the graphed based classification routing presented in [39] leverage semi-supervised data (data with labels and without labels) and creates a graph structure of the data. Then, it estimates the edge weights using a modified LLE method followed by performing a classification which is done based edge weights. However, using a LDR method such as LLE for nonlinear data is not very efficient as it does not leverage the nonlinear features of the data. We believe that replacing the LLE routing in this method by SGE can produce improved results in many such datasets.

The NDR method that we introduced here ensures better performance and preserves the topology of the manifold by emphasizing the smoothness of the manifold when embedding sparse and noisy data. This method is an extension of famous NDR method Isomap where we replaced the geodesics with smoothing splines. In the future, we plan to examine such techniques in more generally. For example, one can imagine generalizing Isomap to the case where both geodesics and smoothing splines are not a good approximation of long manifold distances. In such a case, one can attempt to treat the long manifold distances as unknown, and employ matrix completion techniques like [43,44] on distance matrices where some entries are not observed.

Acknowledgement

The authors would like to thank NSF XSEDE for allocating computational time, under the request number DMS180007, from the resource Jetstream.

References

[1] S. Aksoy, N.H. Younan, L. Bruzzone, Pattern recognition in remote sensing, Pattern Recognit. Lett. 31 (10) (2010) 1069–1070, doi:10.1016/j.patrec.2010.04.014.

[2] C.H. Yu, D. Luo, W. Ding, J. Cohen, D. Small, S. Islam, Spatio-temporal asynchronous co-occurrence pattern for big climate data towards long-lead flood prediction, in: IEEE International Conference on Big Data, 2015, pp. 865–870, doi:10.1109/BigData.2015.7363834.

[3] B.S. Manjunath, W.Y. Ma, Texture features for browsing and retrieval of large image data, IEEE Trans. Pattern Anal. Mach. Intell. 18 (8) (1996) 837–842, doi:10.1109/34.531803.

[4] W. Jin, L. Wang, X. Zeng, Z. Liu, R. Fu, Classification of clouds in satellite imagery using over-complete dictionary via sparse representation, Pattern Recognit. Lett. 49 (Supplement C) (2014) S193–S200, doi:10.1016/j.patrec.2014.07.015.

[5] R. Chaker, Z.A. Aghbari, I.N. Junejo, Social network model for crowd anomaly detection and localization, Pattern Recognit. 61 (Supplement C) (2016) 266–281, doi:10.1016/j.patcog.2016.06.016.

[6] M.S. Parwez, D.B. Rawat, M. Garuba, Big data analytics for user-activity analysis and user-anomaly detection in mobile wireless network, IEEE Trans. Ind. Inf. 13 (4) (2017) 2058–2065, doi:10.1109/TII.2017.2650206.

[7] A.W.C. Liew, H. Yan, M. Yang, Pattern recognition techniques for the emerging field of bioinformatics: a review, Pattern Recognit. 38 (2005) (2006) 2055–2073, doi:10.1016/j.patcog.2005.02.019.

[8] I. Berg, D. Bosnacki, P.A.J. Hilbers, Large scale analysis of small repeats via mining of the human genome, in: 20th International Workshop on Database and Expert Systems Application, 2009, pp. 198–202, doi:10.1109/DEXA.2009.78.

[9] Y. Kong, Y. Jia, Y. Fu, Interactive phrases: semantic descriptions for human interaction recognition, IEEE Trans. Pattern Anal. Mach. Intell. 36 (9) (2014) 1775–1788, doi:10.1109/TPAMI.2014.2303090.

[10] B. Solmaz, B.E. Moore, M. Shah, Identifying behaviors in crowded scenes using stability analysis for dynamical systems, IEEE Trans. Pattern Anal. Mach. Intell. 34 (10) (2012) 1–8, doi:10.1109/TPAMI.2012.123.

[11] L. Van Der Maaten, E. Postma, J. Den Herik, Dimensionality reduction: a comparative, J. Mach. Learn. Res. 10 (2009) 66–71. doi: 10.1.1.112.5472. http://citeseerx.ist.psu.edu/viewdoc/summary?.

[12] I. Jolliffe, Principal Component Analysis, Wiley Online Library, 2002.

[13] T.F. Cox, M.A.A. Cox, Multidimensional Scaling, CRC press, 2000.

[14] J.B. Tenenbaum, V. De Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (5500) (2000) 2319–2323, doi:10.1126/science.290.5500.2319.

[15] P. Delellis, G. Polverino, G. Ustuner, N. Abaid, S. Macrì, E.M. Bollt, M. Porfiri, Collective behaviour across animal species, Sci. Rep. 4 (1) (2014) 3723, doi:10.1038/srep03723.

[16] M.H. Yang, Face recognition using extended isomap, in: IEEE International Conference on Image Processing, 2, 2002, pp. 117–120, doi:10.1109/ICIP.2002.1039901.

[17] M.H. Yang, Extended isomap for classification, in: Proceedings of the National Conference on Artificial Intelligence, 2002, pp. 224–229, doi:10.1109/ICPR.2002.1048014.

[18] M. Balasubramanian, The isomap algorithm and topological stability, Science 295 (5552) (2002) 7, doi:10.1126/science.295.5552.7a.

[19] J.A. Lee, A. Lendasse, M. Verleysen, Nonlinear projection with curvilinear distances: isomap versus curvilinear distance analysis, Neurocomputing 57 (2004) 49–76, doi:10.1016/j.neucom.2004.01.007.

[20] H. Zha, Z. Zhang, Continuum isomap for manifold learnings, Comput. Stat. Data Anal. 52 (1) (2007) 184–200, doi:10.1016/j.csda.2006.11.027.

[21] J.H. Freidman, J.L. Bentley, R.A. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Trans. Math. Softw. 3 (3) (1977) 209–226, doi:10.1145/355744.355745.

[22] P.K. Agarwal, J. Erickson, Geometric range searching and its relatives, Adv. Discr. Computat. Geom. 223 (1997) 1–56. doi: 10.1.1.38.6261.

[23] R.W. Floyd, Algorithm 97: shortest path, Commun. ACM 5 (6) (1962) 345, doi:10.1145/367766.368168.

[24] S. Xiang, F. Nie, C. Zhang, C. Zhang, Nonlinear dimensionality reduction with local spline embedding, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1285–1298, doi:10.1109/TKDE.2008.204.

[25] K. Gajamannage, S. Butail, M. Porfiri, E.M. Bollt, Dimensionality reduction of collective motion by principal manifolds, Physica D 291 (2015) 62–73, doi:10.1016/j.physd.2014.09.009.

[26] J.A. Lee, M. Verleysen, J.A. Lee, M. Verleysen, J.A. Lee, M. Verleysen, Nonlinear Dimensionality Reduction, Springer Science & Business Media, 2007, doi:10.1007/978-0-387-39351-3. https://books.google.com/books?hl=en&lr=&id=o_TIoyeO7AsC&oi=fnd&pg=PR14&dq=nonlinear+dimension+reduction&ots=CNL84oi4Bz&sig=mHZtuSiVafamqZobWXYCbZA95to http://books.google.com/books?hl=en&lr=&id=o_TIoyeO7AsC&oi=fnd&pg=PR14&dq=nonlinear+dimensionality+redu.

[27] E.W. Dijkstra, A note on two problems in connexion with graphs, Numer. Math. 1 (1) (1959) 269–271, doi:10.1007/BF01386390.

[28] S.S. Ray, Graph Theory with Algorithms and its Applications: In Applied Science and Technology, Springer Science & Business Media, 2012.

[29] C.D. Boor, On calculating with B-spline, J. Approx. Theory 6 (1) (1972) 50–62, doi:10.1016/0021-9045(72)90080-9.

[30] C.H. Reinsch, Smoothing by spline functions, Numer.Math. 10 (3) (1967) 177–183, doi:10.1007/BF02162161.

[31] M. Bernstein, V. De Silva, J.C. Langford, J.B. Tenenbaum, Graph approximations to geodesics on embedded manifolds, Technical Report, Department of Psychology, Stanford University, 2000. doi: 10.1.1.32.6460. http://citeseerx.ist.psu.edu/viewdoc/summary?.

[32] J.B. Tenenbaum, V. De Silva, J.C. Langford, Data sets for nonlinear dimensionality reduction, Data for faces, 2016, http://web.mit.edu/cocosci/isomap/datasets.html.

[33] Y. LeCun, C. Cortes, C.J.C. Burges, The MNIST database of handwritten digits(2016). http://yann.lecun.com/exdb/mnist/.

[34] J. Stewart, Essential Calculus: Early Transcendentals, Cengage Learning, 2012.

[35] J.D. Petruccelli, B. Nandram, M. Chen, Applied Statistics for Engineers and Scientists, Prentice Hall New Jersey, 1999.

[36] B. Shaw, T. Jebara, Structure preserving embedding, in: Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09, 2009, pp. 1–8, doi:10.1145/1553374.1553494.

[37] R. Balakrishnan, K. Ranganathan, A Textbook of Graph Theory, Springer Science & Business Media, 2012, doi:10.1007/978-1-4614-4529-6.

[38] T.H. Cormen, Introduction to Algorithms, MIT press, 2009.

[39] J. Wang, F. Wang, C. Zhang, H.C. Shen, L. Quan, Linear neighborhood propagation and its applications, IEEE Trans. Pattern Anal. Mach. Intell. 31 (9) (2009) 1600–1615, doi:10.1109/TPAMI.2008.216. http://ieeexplore.ieee.org/abstract/document/4620115/.

[40] F. Wang, J. Sun, PSF: A unified patient similarity evaluation framework through metric learning with weak supervision, IEEE J. Biomed. Health Inform. 19 (3) (2015) 1053–1060, doi:10.1109/JBHI.2015.2425365. http://ieeexplore.ieee.org/abstract/document/7091853/.

[41] S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.

[42] M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, Neural Comput. 15 (6) (2003) 1373–1396, doi:10.1162/089976603321780317. http://www.mitpressjournals.org/doi/10.1162/089976603321780317.

[43] Z. Lin, M. Chen, Y. Ma, The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices, arXiv preprint (2010).

[44] R. Paffenroth, P. du Toit, R. Nong, L. Scharf, A.P. Jayasumana, V. Bandara, Space-time signal processing for distributed pattern detection in sensor networks, IEEE J. Sel. Top. Signal Process. 7 (1) (2013) 38–49, doi:10.1109/JSTSP.2012.2237381.

Dr. Kelum Gajamannage is a PostDoctoral Scholar in the Department of Mathematical Sciences, WPI, USA. He was awarded his BS in Mathematics and MS in Applied Statistics at University of Peradeniya, and PhD in Mathematics at Clarkson University. His research interests include manifold learning, dimensionality reduction, and data mining.

Dr. Randy Paffenroth is an Associate Professor of Mathematical Sciences, Computer Science, and Data Science at WPI. His current technical interests include machine learning, signal processing, large scale data analytics, compressed sensing, and the interaction between mathematics, computer science and software engineering, with a focus on applications in cyber-defense.

Dr. Erik M. Bollt is endowed as the W. Jon Harrington Professor of Mathematics at Clarkson University and joint to ECE. Professor Bollt specializes in dynamical systems, including as informed by data processing. Prof. Bollt has recently published a book on these topics, [Applied and Computational Measurable Dynamics, SIAM, (2013)].